PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-073576
(43)Date of publication of application : 12.03.2002

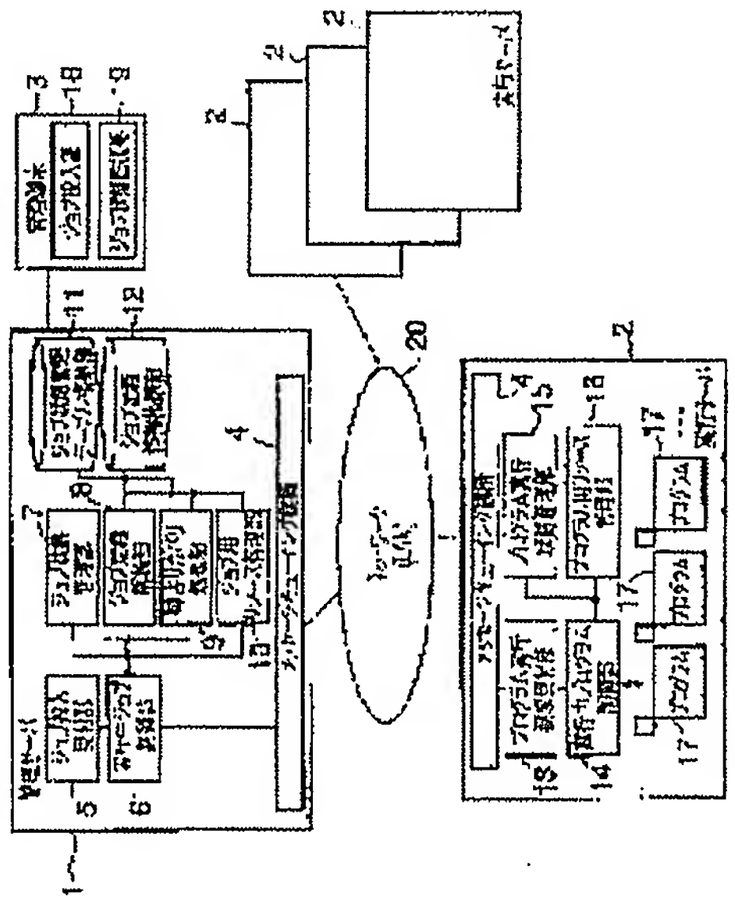
(51)Int.Cl. G06F 15/16
G06F 9/46
G06F 11/00
G06F 15/177

(21)Application number : 2000-262916
(22)Date of filing : 31.08.2000

(71)Applicant : TOSHIBA CORP
(72)Inventor : MATSUO AKIRA

(54) BATCH JOB CONTROL SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a batch job control system which can easily and surely perform a series of processings on a job constituted of a plurality of processing programs on a plurality of computers which are mutually connected through a network.
SOLUTION: The batch job control system is provided with a managing server 1 managing the performing order of the respective processing programs in the job and the computer becoming a performing destination, a plurality of performing servers 2 performing the respective processing programs in the job and a managing terminal 3 connected to the managing server 1. The managing server 1 is connected to each of the performing servers 2 through the network 20. The managing server 1 and the performing servers 2 respectively have message queuing mechanisms 4 for performing asynchronous message communication. The respective processing programs are associated by transferring the performing request of the respective processing programs in the job as a message between the managing server 1 and the performing servers 2 through the message queuing mechanisms 4.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2002－73576
（P2002－73576A）
(43)公開日　平成14年３月12日(2002.3.12)

(51)Int.Cl.[7]		識別記号	FI			テーマコード(参考)
G06F	15/16	620	G06F	15/16	620A	5B045
	9/46	330		9/46	330C	5B098
		360			360B	
	11/00	330		11/00	330A	
	15/177	670		15/177	670F	

審査請求　未請求　請求項の数12　ＯＬ　（全 12 頁）　最終頁に続く

(21)出願番号　特願2000－262916(P2000－262916)

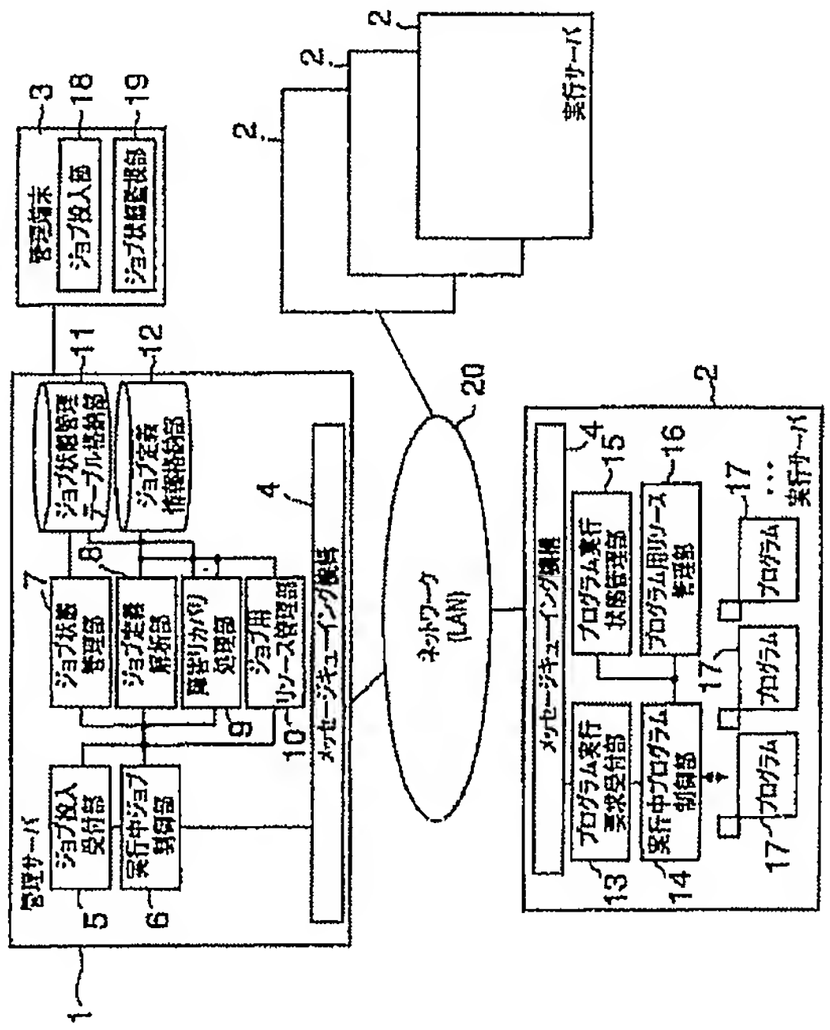
(22)出願日　平成12年８月31日(2000.8.31)

(71)出願人　000003078
株式会社東芝
東京都港区芝浦一丁目１番１号
(72)発明者　松　尾　朗
東京都府中市東芝町１番地　株式会社東芝
府中事業所内
(74)代理人　100064285
弁理士　佐藤　一雄　（外３名）

Ｆターム(参考)　5B045 BB11 BB28 BB34 BB48 BB49
BB53 GC09 HH02 JJ02 JJ13
JJ42
5B098 AA10 GA03 GB13 GC01 GC16
GD01 GD14 JJ03

(54)【発明の名称】　バッチジョブ制御システム

(57)【要約】
【課題】　ネットワークを介して互いに接続された複数の計算機上で複数の処理プログラムからなるジョブの一連の処理を容易かつ確実に実行することができるバッチジョブ制御システムを提供する。
【解決手段】　バッチジョブ制御システムは、ジョブ中の各処理プログラムの実行順序および実行先となる計算機を管理する管理サーバ1と、ジョブ中の各処理プログラムを実行する複数の実行サーバ2と、管理サーバ1に接続された管理端末3とを備えている。管理サーバ1と各実行サーバ2とはネットワーク２０を介して接続されている。管理サーバ1および各実行サーバ2はそれぞれ非同期のメッセージ通信を行うためのメッセージキューイング機構4を有し、管理サーバ1と各実行サーバ2との間でメッセージキューイング機構4を介してジョブ中の各処理プログラムの実行要求をメッセージとしてやりとりすることにより各処理プログラムを連携させるようになっている。



1

【特許請求の範囲】

【請求項1】複数の処理プログラムからなるジョブの実行を制御し、当該各処理プログラムを連携させて一連の処理を実行するバッチジョブ制御システムにおいて、

ジョブ中の各処理プログラムの実行順序および実行先となる計算機を管理する管理計算機と、

前記管理計算機に接続され、ジョブ中の各処理プログラムを実行する複数の実行計算機とを備え、

前記管理計算機および前記各実行計算機はそれぞれ非同期のメッセージ通信を行うためのメッセージキューイング手段を有し、前記管理計算機と前記各実行計算機との間で前記メッセージキューイング手段を介してジョブ中の各処理プログラムの実行要求をメッセージとしてやりとりすることにより前記各処理プログラムを連携させることを特徴とするバッチジョブ制御システム。

【請求項2】前記管理計算機は、ジョブの実行要求を受け付けるジョブ投入受付手段と、

ジョブ中の各処理プログラムの実行順序および実行先となる計算機に関するジョブ定義情報を格納するジョブ定義情報格納手段と、

前記ジョブ定義情報格納手段に格納されたジョブ定義情報を解析し、前記ジョブ投入受付手段により受け付けられたジョブについて、当該ジョブ中の各処理プログラムの実行順序および実行先となる計算機を決定するジョブ定義解析手段と、

前記ジョブ投入受付手段により受け付けられたジョブの実行を制御し、前記ジョブ定義解析手段により決定された実行順序に基づいて、該当する実行計算機に対して該当する処理プログラムの実行要求をメッセージとして送信する実行中ジョブ制御手段とを有し、

前記各実行計算機は、ジョブ中の各処理プログラムの実行要求のメッセージを受け付けるプログラム実行要求受付手段と、

前記プログラム実行要求受付手段により受け付けられた実行要求に対応する処理プログラムを実行し、その実行終了通知を前記管理計算機に対してメッセージとして送信する実行中プログラム制御手段とを有することを特徴とする請求項1記載のバッチジョブ制御システム。

【請求項3】前記管理計算機は、実行中のジョブの進行状態を管理するジョブ状態管理手段をさらに有し、

前記各実行計算機は、実行中の処理プログラムの進行状態を管理するプログラム実行状態管理手段をさらに有することを特徴とする請求項2記載のバッチジョブ制御システム。

【請求項4】前記管理計算機の前記ジョブ定義情報格納手段は、前記ジョブ定義情報の一部として、ジョブまたはジョブ中の各処理プログラムのリカバリ方法に関する情報をさらに格納し、

前記管理計算機は、ジョブの実行中に障害が発生したときに前記ジョブ定義情報格納手段に格納された前記リカバリ方法に関する情報に基づいてジョブまたはジョブ中の各処理プログラムの復旧処理を行う障害リカバリ処理手段をさらに有することを特徴とする請求項2記載のバッチジョブ制御システム。

2

【請求項5】前記管理計算機の前記ジョブ定義情報格納手段は、前記ジョブ定義情報の一部として、ジョブを実行する上で必要とされる計算機リソースに関する情報をさらに格納し、

前記管理計算機は、前記ジョブ定義情報格納手段に格納された前記計算機リソースに関する情報に基づいてジョブを実行する上で必要とされる計算機リソースをジョブ単位で管理するジョブ用リソース管理手段をさらに有することを特徴とする請求項2記載のバッチジョブ制御システム。

【請求項6】前記管理計算機の前記ジョブ定義情報格納手段は、前記ジョブ定義情報の一部として、ジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースに関する情報をさらに格納し、

前記管理計算機は、前記各実行計算機に対してジョブ中の各処理プログラムの実行要求とともに前記計算機リソースに関する情報を送信し、

前記各実行計算機は、前記管理計算機から送信された前記計算機リソースに関する情報に基づいてジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースを処理プログラム単位で管理するプログラム用リソース管理手段をさらに有することを特徴とする請求項2記載のバッチジョブ制御システム。

【請求項7】複数の処理プログラムからなるジョブの実行を制御し、当該各処理プログラムを連携させて一連の処理を実行するバッチジョブ制御システムにおいて、

ジョブ中の各処理プログラムを実行する複数の実行計算機を備え、

前記各実行計算機はそれぞれ非同期のメッセージ通信を行うためのメッセージキューイング手段を有し、前記各実行計算機の間で前記メッセージキューイング手段を介してジョブの実行要求をメッセージとしてやりとりすることにより前記各処理プログラムを連携させることを特徴とするバッチジョブ制御システム。

【請求項8】前記各実行計算機は、ジョブの実行要求として、ジョブ中の各処理プログラムの実行順序および実行先となる計算機に関するジョブ定義情報と実行中のジョブの進行状態に関する情報とを受け付けるジョブ実行要求受付手段と、

前記ジョブ実行要求受付手段により受け付けられたジョブについて、前記ジョブ定義情報と前記実行中のジョブの進行状態に関する情報とを解析し、ジョブ中の未実行の処理プログラムの実行順序および実行先となる計算機を決定するジョブ定義解析手段と、

前記ジョブ投入受付手段により受け付けられたジョブの実行を制御し、前記ジョブ定義解析手段により決定され

たジョブ中の未実行の処理プログラムの実行順序に基づいて、該当する実行計算機に対して未実行の処理プログラムを含むジョブの実行要求として前記ジョブ定義情報と前記実行中のジョブの進行状態に関する情報とをメッセージとして送信する実行中ジョブ制御手段と、
前記ジョブ実行要求受付手段により受け付けられたジョブ中の処理プログラムのうち、当該実行計算機で実行すべき処理プログラムを実行する実行中プログラム制御手段とを有することを特徴とする請求項7 記載のバッチジョブ制御システム。
【 請求項9 】前記各実行計算機は、実行中のジョブの進行状態を管理するジョブ状態管理手段と、
実行中の処理プログラムの進行状態を管理するプログラム実行状態管理手段とをさらに有することを特徴とする請求項8 記載のバッチジョブ制御システム。
【 請求項１０】前記ジョブ定義情報は、ジョブまたはジョブ中の各処理プログラムのリカバリ方法に関する情報をさらに含み、
前記各実行計算機は、ジョブの実行中に障害が発生したときに前記ジョブ定義情報に含まれる前記リカバリ方法に関する情報に基づいてジョブまたはジョブ中の各処理プログラムの復旧処理を行う障害リカバリ処理手段をさらに有することを特徴とする請求項8 記載のバッチジョブ制御システム。
【 請求項１１】前記ジョブ定義情報は、ジョブを実行する上で必要とされる計算機リソースに関する情報をさらに含み、
前記各実行計算機は、前記ジョブ定義情報に含まれる前記計算機リソースに関する情報に基づいてジョブを実行する上で必要とされる計算機リソースをジョブ単位で管理するジョブ用リソース管理手段をさらに有することを特徴とする請求項8 記載のバッチジョブ制御システム。
【 請求項１２】前記ジョブ定義情報は、ジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースに関する情報をさらに含み、
前記各実行計算機は、前記ジョブ定義情報に含まれる前記計算機リソースに関する情報に基づいてジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースを処理プログラム単位で管理するプログラム用リソース管理手段をさらに有することを特徴とする請求項8 記載のバッチジョブ制御システム。
【 発明の詳細な説明】
【０００１】
【 発明の属する技術分野】本発明は、複数の処理プログラムからなるジョブの実行を制御するバッチジョブ制御システムに係り、とりわけ、ネットワークを介して互いに接続された複数の計算機の間でジョブ中の各処理プログラムを連携させて一連の処理を実行するバッチジョブ制御システムに関する。
【０００２】

【 従来の技術】従来から、複数の処理プログラムからなるジョブの実行を制御するシステムとして、ジョブ中の複数の処理プログラムを単一の計算機上で順次実行するバッチジョブ制御システムが知られている。
【０００３】ところで、近年、ネットワークの普及に伴って複数の計算機が互いに接続された分散システムが広く用いられるようになってきており、このような分散システム上でジョブ中の各処理プログラムを連携させて一連の処理を実行することが望まれている。
【０００４】
【 発明が解決しようとする課題】しかしながら、上述したような分散システムでは、複数の計算機がネットワークを介して互いに接続されていることから、これらの計算機の間で処理時間や処理タイミング等が異なる複数の処理プログラムを連携させることが困難であるという問題がある。
【０００５】また、上述したような分散システムでは、オープン環境下でプラットフォーム（ＯＳ等）が異なる複数の計算機が混在して用いられることが多く、これらの計算機の間で適切にデータをやりとりすることが困難であるという問題がある。
【０００６】さらに、上述したような分散システムでは、複数の計算機の間で大量のデータの処理および制御を行う必要があり、ネットワークや計算機等の障害に対して、計算機の間でやりとりされるデータの処理および制御の流れを保証して信頼性を保つことが困難であるという問題がある。
【０００７】本発明はこのような点を考慮してなされたものであり、ネットワークを介して互いに接続された複数の計算機上で複数の処理プログラムからなるジョブの一連の処理を容易かつ確実に実行することができるバッチジョブ制御システムを提供することを目的とする。
【０００８】
【 課題を解決するための手段】本発明は、その第1 の解決手段として、複数の処理プログラムからなるジョブの実行を制御し、当該各処理プログラムを連携させて一連の処理を実行するバッチジョブ制御システムにおいて、
ジョブ中の各処理プログラムの実行順序および実行先となる計算機を管理する管理計算機と、前記管理計算機に接続され、ジョブ中の各処理プログラムを実行する複数の実行計算機とを備え、前記管理計算機および前記各実行計算機はそれぞれ非同期のメッセージ通信を行うためのメッセージキューイング手段を有し、前記管理計算機と前記各実行計算機との間で前記メッセージキューイング手段を介してジョブ中の各処理プログラムの実行要求をメッセージとしてやりとりすることにより前記各処理プログラムを連携させることを特徴とするバッチジョブ制御システムを提供する。
【０００９】なお、上述した第1 の解決手段において、
前記管理計算機は、ジョブの実行要求を受け付けるジョ

5

ブ投入受付手段と、ジョブ中の各処理プログラムの実行順序および実行先となる計算機に関するジョブ定義情報を格納するジョブ定義情報格納手段と、前記ジョブ定義情報格納手段に格納されたジョブ定義情報を解析し、前記ジョブ投入受付手段により受け付けられたジョブについて、当該ジョブ中の各処理プログラムの実行順序および実行先となる計算機を決定するジョブ定義解析手段と、前記ジョブ投入受付手段により受け付けられたジョブの実行を制御し、前記ジョブ定義解析手段により決定された実行順序に基づいて、該当する実行計算機に対して該当する処理プログラムの実行要求をメッセージとして送信する実行中ジョブ制御手段とを有し、前記各実行計算機は、ジョブ中の各処理プログラムの実行要求のメッセージを受け付けるプログラム実行要求受付手段と、前記プログラム実行要求受付手段により受け付けられた実行要求に対応する処理プログラムを実行し、その実行終了通知を前記管理計算機に対してメッセージとして送信する実行中プログラム制御手段とを有することが好ましい。

【００１０】また、上述した第1の解決手段において、前記管理計算機は、実行中のジョブの進行状態を管理するジョブ状態管理手段をさらに有し、前記各実行計算機は、実行中の処理プログラムの進行状態を管理するプログラム実行状態管理手段をさらに有することが好ましい。また、前記管理計算機の前記ジョブ定義情報格納手段は、前記ジョブ定義情報の一部として、ジョブまたはジョブ中の各処理プログラムのリカバリ方法に関する情報をさらに格納し、前記管理計算機は、ジョブの実行中に障害が発生したときに前記ジョブ定義情報格納手段に格納された前記リカバリ方法に関する情報に基づいてジョブまたはジョブ中の各処理プログラムの復旧処理を行う障害リカバリ処理手段をさらに有することが好ましい。さらに、前記管理計算機の前記ジョブ定義情報格納手段は、前記ジョブ定義情報の一部として、ジョブを実行する上で必要とされる計算機リソースに関する情報をさらに格納し、前記管理計算機は、前記ジョブ定義情報格納手段に格納された前記計算機リソースに関する情報に基づいてジョブを実行する上で必要とされる計算機リソースをジョブ単位で管理するジョブ用リソース管理手段をさらに有することが好ましい。さらにまた、前記管理計算機の前記ジョブ定義情報格納手段は、前記ジョブ定義情報の一部として、ジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースに関する情報をさらに格納し、前記管理計算機は、前記各実行計算機に対してジョブ中の各処理プログラムの実行要求とともに前記計算機リソースに関する情報を送信し、前記各実行計算機は、前記管理計算機から送信された前記計算機リソースに関する情報に基づいてジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースを処理プログラム単位で管理するプログラム用リソース管理

6

手段をさらに有することが好ましい。

【００１１】本発明は、その第2の解決手段として、複数の処理プログラムからなるジョブの実行を制御し、当該各処理プログラムを連携させて一連の処理を実行するバッチジョブ制御システムにおいて、ジョブ中の各処理プログラムを実行する複数の実行計算機を備え、前記各実行計算機はそれぞれ非同期のメッセージ通信を行うためのメッセージキューイング手段を有し、前記各実行計算機の間で前記メッセージキューイング手段を介してジョブの実行要求をメッセージとしてやりとりすることにより前記各処理プログラムを連携させることを特徴とするバッチジョブ制御システムを提供する。

【００１２】なお、上述した第2の解決手段において、前記各実行計算機は、ジョブの実行要求として、ジョブ中の各処理プログラムの実行順序および実行先となる計算機に関するジョブ定義情報と実行中のジョブの進行状態に関する情報とを受け付けるジョブ実行要求受付手段と、前記ジョブ実行要求受付手段により受け付けられたジョブについて、前記ジョブ定義情報と前記実行中のジョブの進行状態に関する情報とを解析し、ジョブ中の未実行の処理プログラムの実行順序および実行先となる計算機を決定するジョブ定義解析手段と、前記ジョブ投入受付手段により受け付けられたジョブの実行を制御し、前記ジョブ定義解析手段により決定されたジョブ中の未実行の処理プログラムの実行順序に基づいて、該当する実行計算機に対して未実行の処理プログラムを含むジョブの実行要求として前記ジョブ定義情報と前記実行中のジョブの進行状態に関する情報とをメッセージとして送信する実行中ジョブ制御手段と、前記ジョブ実行要求受付手段により受け付けられたジョブ中の処理プログラムのうち、当該実行計算機で実行すべき処理プログラムを実行する実行中プログラム制御手段とを有することが好ましい。

【００１３】また、上述した第2の解決手段において、前記各実行計算機は、実行中のジョブの進行状態を管理するジョブ状態管理手段と、実行中の処理プログラムの進行状態を管理するプログラム実行状態管理手段とをさらに有することが好ましい。また、前記ジョブ定義情報は、ジョブまたはジョブ中の各処理プログラムのリカバリ方法に関する情報をさらに含み、前記各実行計算機は、ジョブの実行中に障害が発生したときに前記ジョブ定義情報に含まれる前記リカバリ方法に関する情報に基づいてジョブまたはジョブ中の各処理プログラムの復旧処理を行う障害リカバリ処理手段をさらに有することが好ましい。さらに、前記ジョブ定義情報は、ジョブを実行する上で必要とされる計算機リソースに関する情報をさらに含み、前記各実行計算機は、前記ジョブ定義情報に含まれる前記計算機リソースに関する情報に基づいてジョブを実行する上で必要とされる計算機リソースをジョブ単位で管理するジョブ用リソース管理手段をさらに

有することが好ましい。さらにまた、前記ジョブ定義情報は、ジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースに関する情報をさらに含み、前記各実行計算機は、前記ジョブ定義情報に含まれる前記計算機リソースに関する情報に基づいてジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースを処理プログラム単位で管理するプログラム用リソース管理手段をさらに有することが好ましい。

【００１４】本発明の第1および第2の解決手段によれば、複数の計算機のそれぞれに非同期のメッセージ通信を行うためのメッセージキューイング機構4を設け、メッセージキューイング機構4を介してジョブ中の各処理プログラムの実行要求またはジョブの実行要求をメッセージとしてやりとりすることにより各処理プログラムを連携させているので、ネットワークを介して互いに接続された複数の計算機上で複数の処理プログラムからなるジョブの一連の処理を容易かつ確実に実行することができる。

【００１５】

【発明の実施の形態】以下、図面を参照して本発明の実施の形態について説明する。

【００１６】第1の実施の形態

図1乃至図5は本発明によるバッチジョブ制御システムの第1の実施の形態を説明するための図である。

【００１７】まず、図1により、本発明の第1の実施の形態に係るバッチジョブ制御システムの全体構成について説明する。図1に示すように、バッチジョブ制御システムは、複数のバッチ処理プログラム（以下単に「処理プログラム」ともいう）からなるジョブの実行を制御し、当該各処理プログラムを連携させて一連の処理を実行するものである。バッチジョブ制御システムは、ジョブ中の各処理プログラムの実行順序および実行先となる計算機を管理する管理サーバ（管理計算機）1と、管理サーバ1に接続され、ジョブ中の各処理プログラムを実行する複数の実行サーバ（実行計算機）2と、管理サーバ1に接続された管理端末3とを備えている。

【００１８】図1において、管理サーバ1と各実行サーバ2とはＬＡＮ等のネットワーク２０を介して接続されている。また、管理サーバ1および各実行サーバ2はそれぞれ非同期のメッセージ通信を行うためのメッセージキューイング機構4を有し、管理サーバ1と各実行サーバ2との間でメッセージキューイング機構4を介してジョブ中の各処理プログラムの実行要求をメッセージとしてやりとりすることにより各処理プログラムを連携させるようになっている。

【００１９】ここで、管理サーバ1は、ジョブの実行要求を受け付けるジョブ投入受付部5と、ジョブ中の各処理プログラムの実行順序および実行先となる計算機に関するジョブ定義情報を格納するジョブ定義情報格納部12とを有している。なお、ジョブ定義情報には、ジョブ中の各処理プログラムの実行順序および実行先となる計算機に関する情報に加えて、ジョブまたはジョブ中の各処理プログラムのリカバリ方法に関する情報と、ジョブまたはジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースに関する情報とが格納されている。なお、ジョブ定義情報の詳細については後述する。

【００２０】また、管理サーバ1は、ジョブ定義情報格納部12に格納されたジョブ定義情報を解析し、ジョブ投入受付部5により受け付けられたジョブについて、当該ジョブ中の各処理プログラムの実行順序および実行先となる計算機を決定するジョブ定義解析部8と、ジョブ投入受付部5により受け付けられたジョブの実行を制御し、ジョブ定義解析部8により決定された実行順序に基づいて、該当する実行サーバ2に対して該当する処理プログラムの実行要求をメッセージとして送信する実行中ジョブ制御部6とを有している。ここで、管理サーバ1の実行中ジョブ制御部6は、実行サーバ2に対して処理プログラムの実行要求をメッセージとして送信した後、実行サーバ2から処理プログラムの実行終了通知を待ち、当該実行終了通知を受信した時点で、次の処理プログラムの実行要求をメッセージとして送信し、最後の処理プログラムの実行が終了するまで同様の処理を繰り返すようになっている。なお、管理サーバ1の実行中ジョブ制御部6から送信されたメッセージは管理サーバ1のメッセージキューイング機構4を介して各実行サーバ2に対して送信される。

【００２１】さらに、管理サーバ1は、実行中のジョブの進行状態を表すジョブ状態管理テーブルを格納するジョブ状態管理テーブル格納部11と、ジョブ状態管理テーブル格納部11に格納されたジョブ状態管理テーブルを用いて実行中のジョブの進行状態を管理するジョブ状態管理部7とを有している。なお、ジョブ状態管理テーブルには、ジョブ中のどの処理プログラムが現在どの計算機上で実行されているかという情報の他、既に終了したジョブまたはジョブ中の処理プログラムのステータス情報が格納されており、実行中ジョブ制御部6からの要求に基づいてジョブ状態管理部7により更新されるようになっている。なお、ジョブ状態管理テーブルの詳細については後述する。

【００２２】さらにまた、管理サーバ1は、ジョブの実行中に障害が発生したときにジョブ定義情報格納部12に格納されたリカバリ方法に関する情報に基づいてジョブまたはジョブ中の各処理プログラムの復旧処理を行う障害リカバリ処理部9と、ジョブ定義情報格納部12に格納された計算機リソースに関する情報に基づいてジョブを実行する上で必要とされる計算機リソースをジョブ単位で管理するジョブ用リソース管理部10とを有している。なお、ジョブ用リソース管理部10は、メモリやＣＰＵ、ディスク容量といった種々の計算機リソースの管理（確保および開放）を行うものであり、実行要求が

9

受け付けられたジョブの実行を開始する際に、ジョブ定義情報格納部12に格納された計算機リソースに関する情報に基づいて必要な計算機リソースの確保を行うようになっている。

【0023】各実行サーバ2は、管理サーバ1の実行中ジョブ制御部6から送信されたジョブ中の各処理プログラムの実行要求のメッセージを受け付けるプログラム実行要求受付部13と、プログラム実行要求受付部13により受け付けられた実行要求に対応する処理プログラムを実行し、その実行終了通知を管理サーバ1に対してメッセージとして送信する実行中プログラム制御部14とを有している。なお、実行中プログラム制御部14は、プログラム実行要求受付部13で受け付けられた実行要求に対応する処理プログラムの起動から終了までの監視を行い、その終了ステータスを管理サーバ1の実行中ジョブ制御部6に対して送信するようになっている。なお、各実行サーバ2の実行中プログラム制御部14から送信されたメッセージは各実行サーバ2のメッセージキューイング機構4を介して管理サーバ1に対して送信される。

【0024】また、各実行サーバ2は、実行中の処理プログラムの進行状態(実行サーバ2上でどの処理プログラムがいくつ同時に実行されているか等)を管理するプログラム実行状態管理部15と、ジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースを処理プログラム単位で管理するプログラム用リソース管理部16とを有している。なお、プログラム用リソース管理部16は、管理サーバ1のジョブ用リソース管理部10と同様に、メモリやCPU、ディスク容量といった種々の計算機リソースの管理(確保および開放)を行うものであり、各実行サーバ2上で処理プログラムの実行を開始する際に、管理サーバ1から引き渡された計算機リソースに関する情報に基づいて必要な計算機リソースの確保を行うようになっている。なお、実行サーバ2には、実際に起動される複数の処理プログラム17がインストールされている。

【0025】なお、管理サーバ1および各実行サーバ2のメッセージキューイング機構4は、管理サーバ1と各実行サーバ2との間でやりとりされるメッセージをキューイングするものであり、管理サーバ1と各実行サーバ2との間でメッセージを非同期にやりとりし、送信先となる実行サーバ2に対してメッセージが確実に届くことを保証するようになっている。すなわち、メッセージキューイング機構4は、ネットワーク20や計算機(管理サーバ1および実行サーバ2)等の障害によりメッセージが送信できないときには、そのメッセージをキューイングし、通信可能な状態になった時点で自動的にメッセージを送信する自動リトライ機能を有している。このため、管理サーバ1または各実行サーバ2は、実行中ジョブ制御部6または実行中プログラム制御部14からメッセージキューイング機構4に対してメッセージを引き渡した後は、相手先となる計算機に対するメッセージの到着の確認を待つことなく、管理サーバ1または各実行サーバ2の処理を継続することができる。

10

【0026】管理端末3は、管理サーバ1に接続された利用者(管理者)用の計算機であり、利用者の指示に従って、またはあらかじめ設定された日時に自動的に管理サーバ1に対してジョブの投入(実行要求)を行うジョブ投入部18と、管理サーバ1で管理されている実行中のジョブの進行状態をジョブ状態管理テーブル格納部11に格納されたジョブ状態管理テーブルを参照して監視するジョブ状態監視部19とを有している。

【0027】次に、図1および図2により、このような構成からなる本発明の第1の実施の形態の作用について説明する。図2は図1に示すバッチジョブ制御システムの全体の処理の流れを説明するためのフローチャートである。

【0028】まず、利用者は、実行対象となるジョブごとにジョブ定義情報を作成し、ジョブ定義情報格納部12に格納する。ここで、ジョブ定義情報を作成する作業は、管理端末3上等でGUI等を備えた専用の定義ツールを用いて行うことができる。なお、ジョブ定義情報としては、ジョブ中の各処理プログラムについて、そのプログラム名、実行順序および実行先となる計算機に関する情報の他、ジョブまたはジョブ中の各処理プログラムのリカバリ方法に関する情報や、ジョブまたはジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースに関する情報等が格納されている。

【0029】図3はジョブ定義情報格納部12に格納されたジョブ定義情報の一例を示す図である。

【0030】図3に示すように、ジョブ定義情報12'は、フロー定義マスタファイル12a、実行順序定義ファイル12bおよび処理プログラム属性定義ファイル12cの3つのファイルからなっている。

【0031】このうち、フロー定義マスタファイル12aは、ジョブ定義情報のマスタファイルであり、識別番号("ID=00000001")およびジョブ名("NAME=業務A")の他、実行順序定義ファイル12bのファイル名("FILE=F01.net")が記述されている。

【0032】実行順序定義ファイル12bは、ジョブ中の各処理プログラムの実行順序を定義したものであり、各処理プログラムの識別番号と、それに先行する処理プログラムの識別番号とが複数組記述されている。

【0033】なお、図3に示す実行順序定義ファイル12bでは、識別番号0010の処理プログラムを先頭として、識別番号0010,0011の処理プログラムを逐次実行した後、識別番号0012,0013の処理プログラムを平行して実行し、最後に、識別番号0012,0013の処理プログラムの終了を待ち合わせて識別番号0014の処理プログラムを実行する場合の記述

例が示されている。

【００３４】図4は図3に示す実行順序定義ファイル１２ｂに従ったジョブの流れを示す模式図である。図4に示すように、ジョブの流れは、(1)識別番号００１０，００１１の処理プログラムのように、先行する処理プログラムの終了を待って次の処理プログラムの実行を開始するタイプ（逐次型）、(2)識別番号００１２，００１３の処理プログラムのように、同時に複数の処理プログラムを平行して実行するタイプ（分岐型）、(3)識別番号００１４の処理プログラムのように、先行する複数の処理プログラムの終了を待ち合わせて次の処理プログラムの実行を開始するタイプ（合流型）の組合せとして表現することができる。

【００３５】処理プログラム属性定義ファイル１２ｃは、ジョブ中の各処理プログラムの属性情報を定義したものであり、識別番号（“ ID=0010”等）、プログラム名（“ NAME=処理A”等）、プログラムのパス（“ PATH=C:/.../p10.bat”等）および実行先となる計算機名（“ HOST=SERVER-A”等）の他、ジョブまたはジョブ中の各処理プログラムのリカバリ方法の名称（“ RECOVERY=リカバリA”等）、およびジョブまたはジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースに関する情報（図示せず）が記述されている。

【００３６】以上のようにしてジョブ定義情報を作成した後、利用者は、管理端末3のジョブ投入部１８を用いてジョブの投入（実行要求）を行う。ここで、実行要求を行うことができるジョブは、あらかじめジョブ定義情報格納部１２にジョブ定義情報が格納されているジョブのみである。なお、ジョブの実行要求は、利用者の指示によって行う他、あらかじめ起動スケジュール（開始日時および開始対象となるジョブ）を設定しておき、その設定された日時に自動的に行うようにすることもできる。

【００３７】図2に示すように、利用者の指示に従って、またはあらかじめ設定された日時に自動的にジョブ投入部１８から出されたジョブの実行要求は、管理サーバ1のジョブ投入受付部5で受け付けられ（ステップ１０１）、実行中ジョブ制御部6へ引き渡される。実行中ジョブ制御部6は、ジョブ定義解析部8に対して問い合わせを行い、ジョブ定義情報格納部１２に格納されたジョブ定義情報（図3のフロー属性定義マスタファイル１２ａおよび実行順序定義ファイル１２ｂ）を読み込み（ステップ１０２）、実行要求が出されたジョブの先頭の処理プログラムを抽出するとともにその処理プログラムの実行先となる計算機を決定する（ステップ１０３）。

【００３８】その後、実行中ジョブ制御部6は、該当する処理プログラムの実行要求をメッセージとしてメッセージキューイング機構4へ引き渡し、該当する実行サーバ2に対してメッセージを送信する（ステップ１０４）。メッセージキューイング機構4は、実行中ジョブ制御部6から引き渡されたメッセージをキューイングし、該当する実行サーバ2のメッセージキューイング機構4との間でメッセージを非同期にやりとりする。ここで、メッセージキューイング機構4は自動リトライ機能を有しており、ネットワーク２０に障害が発生してメッセージが送信できない場合や、送信先となる実行サーバ2がダウンしてメッセージが送信できない場合等において、障害が発生してから障害から復旧するまでの間に一定間隔で自動的にリトライを試み、通信可能な状態になった時点で自動的にメッセージを送信する。なお、ジョブ中の一つの処理プログラムから複数の処理プログラムへ流れが分岐する場合には、管理サーバ1から複数の実行サーバ2に対してメッセージが同報される。また、管理サーバ1と実行サーバ2との間でやりとりされるメッセージは、図3に示す処理プログラム属性ファイル１２ｃに相当するものであり、プログラム名、プログラムのパス、実行先となる計算機名、オプションの情報（リカバリ方法の名称および計算機リソースに関する情報等）等が含まれている。

【００３９】ここで、管理サーバ1の実行中ジョブ制御部6から送信された処理プログラムの実行要求のメッセージは該当する実行サーバ2のプログラム実行要求受付部１３で受け付けられる（ステップ２０１）。その後、このようにして受け付けられた処理プログラムの実行要求のメッセージは、実行中プログラム制御部１４へ引き渡され、当該メッセージ内に含まれる情報に従って、該当する処理プログラム１７が起動される（ステップ２０２）。なお、実行中プログラム制御部１４においては、同時に複数の異なる処理プログラム１７を起動することも可能であり、その場合には、それぞれの処理プログラムに対応して複数のスレッドが起動される。

【００４０】このようにして処理プログラムが起動されると、実行中プログラム制御部１４は、プログラム実行状態管理部１５に対して実行中の処理プログラムの登録を行う（ステップ２０３）。ここで、プログラム実行状態管理部１５は、処理プログラムがいくつ同時に実行されているかを管理し、処理負荷の大きなプログラムを同時に多く実行させないよう制御する。

【００４１】なお、実行中プログラム制御部１４は、起動された処理プログラムの実行が終了すると、同様にしてプログラム実行状態管理部１５に対して処理プログラムの登録削除要求を行う（ステップ２０４）。また、実行中プログラム制御部１４は、終了した処理プログラムの終了ステータスを取得し、管理サーバ1の実行中ジョブ制御部6に対して処理プログラムの実行終了通知をその終了ステータスとともにメッセージとして送信する（ステップ２０５）。なお、このメッセージは、実行サーバ2のメッセージキューイング機構4を介して管理サーバ1に対して送信される。

13

【００４２】このようにして実行サーバ2から送信された実行終了通知は、管理サーバ1の実行中ジョブ制御部6が受信する（ステップ１０５）。実行中ジョブ制御部6は、受信した実行終了通知を終了ステータスとともにジョブ状態管理部7へ引き渡し、ジョブ状態管理テーブル格納部１１に格納されたジョブ状態管理テーブルを更新する（ステップ１０６）。

【００４３】図5はジョブ状態管理テーブル格納部１１に格納されたジョブ状態管理テーブルの一例を示す図である。

【００４４】図5に示すように、ジョブ状態管理テーブル１１′は、ジョブの進行状態に関する情報（ジョブID、業務名、投入日時、投入ユーザ、終了日時、現在のステータス、実行中ジョブステップ）の他、ジョブ中の各処理プログラムの進行状態に関する情報（プログラムＩＤ、処理名、投入日時、実行開始日時、実行終了日時、実行計算機、終了ステータス）が含まれている。

【００４５】なお、実行中ジョブ制御部6は、１つの処理プログラムの実行が終了すると、再びジョブ定義解析部8に対して問い合わせを行い、当該ジョブのジョブ定義情報中に次の処理プログラムが存在するか否かを判断する（ステップ１０７）。

【００４６】ここで、次の処理プログラムが存在する場合には、当該次の処理プログラムを抽出するとともにその処理プログラムの実行先となる計算機を決定し（ステップ１０８）、上述したステップ１０４乃至１０７の処理を繰り返す。なお、複数の処理プログラムから一つの処理プログラムへ流れが合流する場合には、先行する複数の処理プログラムの必要な実行終了通知が全て揃うまで待ち、全ての実行終了通知が揃った時点で、次の処理プログラムの実行要求をメッセージとして送信する。

【００４７】なお、当該ジョブ中の最後の処理プログラムが終了した時点で、一つのジョブが完了する（ステップ１０９）。

【００４８】ここで、このような一連の処理を伴うジョブの実行中に障害が発生したときには、障害リカバリ処理部9により、ジョブ定義情報格納部１２に格納されたリカバリ方法に関する情報に基づいてジョブまたはジョブ中の各処理プログラムの復旧処理（処理プログラムの再実行や停止、およびジョブの再投入等）を行う。なお、障害リカバリ処理部9は利用者が起動することができる。

【００４９】なお、ジョブの実行中に発生する障害としては例えば、実行中の処理プログラムが何らかの原因によりエラーを起こして終了した場合や、ネットワーク２０や計算機（管理サーバ1および実行サーバ2）等の障害によって途中で中断したジョブが自動リトライを繰り返したのにもかかわらず長時間にわたって停滞してしまう場合等がある。なお、このような場合におけるリカバリ方法はジョブの種類によって異なり、ジョブを再開する方法として、エラーを起こした処理プログラムから再度実行し直す方法や、エラーを起こした処理プログラムをスキップして再開する方法をとることができる。また、ジョブを再投入して最初の処理プログラムから実行し直す方法や、エラーが発生した時点で強制的にジョブまたは処理プログラムを終了させる方法等をとることができる。なお、それぞれのジョブまたはジョブ中の処理プログラムに対してとられるリカバリ方法は、ジョブ定義情報格納部１２に格納されたジョブ定義情報（処理プログラム属性定義ファイル１２ｃ等）に記述される。

14

【００５０】ここで、ジョブを再開する場合には、障害リカバリ処理部9は、ジョブ状態管理テーブル格納部１１に格納されたジョブ状態管理テーブルからジョブ中の各処理プログラムの進行状態に関する情報を取得し、ジョブ定義情報に記述されたリカバリ方法に従って、再開する処理プログラムを決定した後、実行中ジョブ制御部6に対して処理プログラムの再開要求等を出す。なおこのとき、障害リカバリ処理部9は、メッセージキューイング機構4に残されている関係するメッセージを全て削除する。また、ジョブを再投入する場合には、障害リカバリ処理部9は、メッセージキューイング機構4に残されている関係するメッセージを全て削除し、ジョブ投入受付部5に対して再投入の指示を行う。さらに、ジョブを強制的に終了させる場合には、障害リカバリ処理部9は、メッセージキューイング機構4に残されている関係するメッセージを全て削除した後、ジョブ状態管理部7によりジョブ状態管理テーブル格納部１１に格納されたジョブ状態管理テーブル中のジョブのステータスを強制終了の状態に更新する。

【００５１】また、このような一連の処理を伴うジョブを実行する際には、管理サーバ1のジョブ用リソース管理部１０により、ジョブを実行する上で必要とされる計算機リソースをジョブ単位で管理する。このとき、ジョブ用リソース管理部１０は、ジョブの実行を開始および終了する際にあらかじめ必要な計算機リソースの確保および開放を行う。具体的には、ジョブの実行を開始する際には、計算機リソースの確保を行い、確保できない場合にはジョブの実行を待ち合わせ、確保できた時点でジョブを開始する。これに対し、このようなジョブ中の各処理プログラムを実行する際には、各実行サーバ2のプログラム用リソース管理部１６により、ジョブ中の各処理プログラムを実行する上で必要とされる計算機リソースを処理プログラム単位で管理する。このとき、プログラム用リソース管理部１６は、処理プログラムの実行を開始および終了する際にあらかじめ必要な計算機リソースの確保および開放を行う。具体的には、処理プログラムの実行を開始する際には、計算機リソースの確保を行い、確保できない場合には処理プログラムの実行を待ち合わせ、確保できた時点で処理プログラムの実行を開始する。なお、ジョブまたはジョブ中の各処理プログラム

15

を実行する上で必要とされる計算機リソースに関する情報はジョブ単位および処理プログラム単位でジョブ定義情報格納部１２に格納されている。

【００５２】ここで、ジョブ用リソース管理部１０およびプログラム用リソース管理部１６において、計算機リソースを確保できない理由としては、当該計算機リソースが他のジョブや処理プログラムにより既に確保されているという場合が一般的である。この場合には、ジョブ用リソース管理部１０およびプログラム用リソース管理部１６は、当該他のジョブまたは処理プログラムの実行が終了して計算機リソースが開放されるまで当該計算機リソースを確保することができないので、当該ジョブまたは処理プログラムは実行待ちの状態となる。その後、ジョブ用リソース管理部１０およびプログラム用リソース管理部１６は、当該他のジョブまたは処理プログラムの実行が終了して計算機リソースが開放された時点で、実行待ちの状態にあるジョブまたはジョブ中の処理プログラムに対して計算機リソースを割り当て、ジョブまたはジョブ中の処理プログラムの実行を順次開始する。なお、このようにして計算機リソースの排他制御を行うことにより、ジョブまたはジョブ中の各処理プログラムの実行時に必要な計算機リソースが不足してエラーが発生することを抑制することができる。

【００５３】このように本発明の第1の実施の形態によれば、複数の計算機（管理サーバ1および実行サーバ２）のそれぞれに非同期のメッセージ通信を行うためのメッセージキューイング機構4を設け、メッセージキューイング機構4を介してジョブ中の各処理プログラムの実行要求をメッセージとしてやりとりすることにより各処理プログラムを連携させているので、ネットワーク２０を介して互いに接続された複数の計算機上で複数の処理プログラムからなるジョブの一連の処理を容易かつ確実に実行することができる。

【００５４】すなわち、本発明の第1の実施の形態によれば、ジョブ中の各処理プログラムの実行要求をメッセージキューイング機構4を介してやりとりしているので、相手先の計算機の状態にかかわらず、管理サーバ1から各実行サーバ2へのメッセージの送信、各実行サーバ2でのメッセージの受け付けおよび実行、および管理サーバ1での各実行サーバ2からのメッセージの待ち合わせ等を容易に行うことができ、このため、処理時間や処理タイミング等が異なる複数の処理プログラムを容易に連携させることができる。また、管理サーバ1および複数の実行サーバ2としてプラットフォーム（ＯＳ等）が異なる複数の計算機が用いられる場合でも、これらの計算機（管理サーバ1および実行サーバ２）の間で適切にデータをやりとりすることができる。さらに、メッセージキューイング機構4により、ネットワーク２０や実行サーバ2等の障害によりメッセージが送信できないときに、そのメッセージをキューイングし、通信可能な状態になった時点で自動的にリトライを試み、通信可能になった時点でメッセージを送信するので、ネットワーク２０や計算機（管理サーバ1および実行サーバ２）等の障害に対して、計算機の間でやりとりされるデータの処理および制御の流れを保証して信頼性を保つことができる。

16

【００５５】第2の実施の形態

次に、図6により、本発明によるバッチジョブ制御システムの第2の実施の形態について説明する。本発明の第２の実施の形態は、ネットワークを介して接続された計算機が全て実行サーバであり、各実行サーバが図1に示す管理サーバの機能を備えている点を除いて、他は図1に示す第1の実施の形態と略同一である。本発明の第2の実施の形態において、図1に示す第1の実施の形態と同一部分には同一符号を付して詳細な説明は省略する。

【００５６】図6に示すように、バッチジョブ制御システムは、ジョブ中の各処理プログラムを実行する複数の実行サーバ（実行計算機）２１を備えている。各実行サーバ２１はそれぞれ非同期のメッセージ通信を行うためのメッセージキューイング機構4を有し、各実行サーバ２１の間でメッセージキューイング機構4を介してジョブの実行要求をメッセージとしてやりとりすることにより各処理プログラムを連携させるようになっている。

【００５７】ここで、各実行サーバ２１は、ジョブの実行要求として、ジョブ中の各処理プログラムの実行順序および実行先となる計算機に関するジョブ定義情報１２′と、実行中のジョブの進行状態を表すジョブ状態管理テーブル１１′とをやりとりしており、これらの情報に基づいてジョブの実行要求を受け付けた各実行サーバ２１が図1に示す管理サーバ1および実行サーバ2の両方の機能を果たすようになっている。

【００５８】すなわち、各実行サーバ２１は、ジョブ定義情報１２′とジョブ状態管理テーブル１１′とを受け付けるジョブ実行要求受付部２２を有しており、このジョブ実行要求受付部２２で受け付けられたジョブの実行要求が実行中ジョブ制御部6および実行中プログラム１４の両方に送られるようになっている。

【００５９】具体的には、ジョブ実行要求受付部２２によりジョブの実行要求が受け付けられると、ジョブ定義情報１２′とジョブ状態管理テーブル１１′とが実行中ジョブ制御部6に送られ、実行中ジョブ制御部6の制御の下で、ジョブ定義情報１２′とジョブ状態管理テーブル１１′とを解析し、ジョブ中の未実行の処理プログラムの実行順序および実行先となる計算機を決定する。なお、実行中ジョブ制御部6は、ジョブ実行要求受付部２２で受け付けられたジョブの実行を制御し、ジョブ定義解析部8により決定されたジョブ中の未実行の処理プログラムの実行順序に基づいて、該当する実行サーバ2に対して未実行の処理プログラムを含むジョブの実行要求としてジョブ定義情報１２′とジョブ状態管理テーブル

17

１１′とをメッセージとして送信する。

【００６０】一方、ジョブ実行要求受付部２２により受け付けられたジョブ中の処理プログラムのうち、当該実行計算機で実行すべき処理プログラムが実行中プログラム制御部１４により実行される。

【００６１】なお、複数の実行サーバ２１のうちの少なくとも一つは外部からのジョブの投入を受け付けるジョブ投入受付部5を有しており、起点となる実行サーバ２１に対して外部からジョブの実行要求（ジョブ定義情報１２′およびジョブ状態管理テーブル１１′を含む）を行うことができるようになっている。

【００６２】このように本発明の第2の実施の形態によれば、複数の計算機（実行サーバ２）のそれぞれに非同期のメッセージ通信を行うためのメッセージキューイング機構4を設け、メッセージキューイング機構4を介してジョブの実行要求をメッセージとしてやりとりすることにより各処理プログラムを連携させているので、上述した第1の実施の形態と同様に、ネットワーク２０を介して互いに接続された複数の計算機（実行サーバ２）上で複数の処理プログラムからなるジョブの一連の処理を容易かつ確実に実行することができる。

【００６３】なお、上述した第1および第2の実施の形態におけるメッセージキューイング機構4、ジョブ投入受付部5、実行中ジョブ制御部6、ジョブ状態管理部７、ジョブ定義解析部8、障害リカバリ処理部9、ジョブ用リソース管理部１０、プログラム実行要求受付部１３、実行中プログラム制御部１４、プログラム実行状態管理部１５、プログラム用リソース管理部１６、ジョブ投入部１８、ジョブ状態監視部１９およびジョブ実行要求受付部２２はいずれも、コンピュータ上で動作するプログラムとして実現することができる。このようなプログラムは、各種の記録媒体に記録され、コンピュータから読み出されて上述したような処理が行われる。

【００６４】なお、上述した第1および第2の実施の形態で用いられる記録媒体としては、磁気ディスク、フロッピー（登録商標）ディスク、ハードディスク、光ディスク（ＣＤ－ＲＯＭ、ＣＤ－Ｒおよびｄｖｄ等）、光磁気ディスク（ＭＯ等）および半導体メモリ等を含み、プログラムを記録することができ、かつコンピュータ読み取り可能なものであれば、その記録形式はどのようなものでもよい。また、記録媒体としては、ネットワーク上で伝送される際の搬送波等の情報伝達媒体を含む。さらに、記録媒体は、コンピュータと独立したものに限らず、ＬＡＮやインターネット等により伝送されたプログラムをダウンロードして記憶または一時記憶した記録媒体も含まれる。さらにまた、記録媒体は1つであるとは限らず、複数の記録媒体から上述した第1および第2の実施の形態で行われる処理が実現される場合も本発明における記録媒体の概念に含まれる。

【００６５】また、上述した第1および第2の実施の形

18

態で用いられる記録媒体としては、記録媒体からコンピュータにインストールされたプログラムの指示に基づき、コンピュータ上で稼働しているオペレーティングシステム（ＯＳ）、データベース管理ソフトおよびネットワークソフト等の他のプログラム（ミドルウェア等）により、上述した第1および第2の実施の形態で行われる処理の一部を実現するようにしてもよい。

【００６６】

【発明の効果】以上説明したように本発明によれば、ネットワークを介して互いに接続された複数の計算機上で複数の処理プログラムからなるジョブの一連の処理を容易かつ確実に実行することができる。

【図面の簡単な説明】

【図1】本発明によるバッチジョブ制御システムの第1の実施の形態を示す全体構成図。

【図2】図1に示すバッチジョブ制御システムの全体の処理の流れを説明するためのフローチャート。

【図3】図1に示すバッチジョブ制御システムで用いられるジョブ定義情報の一例を示す図。

【図4】図3に示すジョブ定義情報に従ったジョブの流れを示す模式図。

【図5】図1に示すバッチジョブ制御システムで用いられるジョブ状態管理テーブルの一例を示す図。

【図6】本発明によるバッチジョブ制御システムの第2の実施の形態を示す全体構成図。

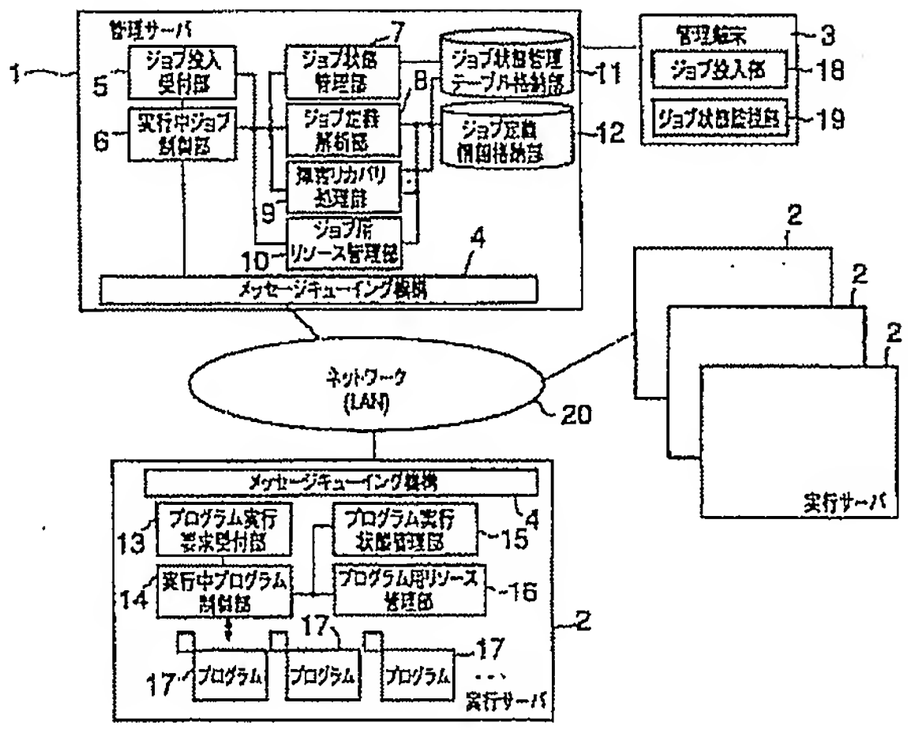
【符号の説明】

１　管理サーバ（管理計算機）
２，２１　実行サーバ（実行計算機）
３　管理端末
４　メッセージキューイング機構
５　ジョブ投入受付部
６　実行中ジョブ制御部
７　ジョブ状態管理部
８　ジョブ定義解析部
９　障害リカバリ処理部
１０　ジョブ用リソース管理部
１１　ジョブ状態管理テーブル格納部
１１′　ジョブ状態管理テーブル
１２　ジョブ定義情報格納部
１２′　ジョブ定義情報
１２ａ　フロー定義マスタファイル
１２ｂ　実行順序定義ファイル
１２ｃ　処理プログラム属性定義ファイル
１３　プログラム実行要求受付部
１４　実行中プログラム制御部
１５　プログラム実行状態管理部
１６　プログラム用リソース管理部
１７　処理プログラム
１８　ジョブ投入部
１９　ジョブ状態監視部

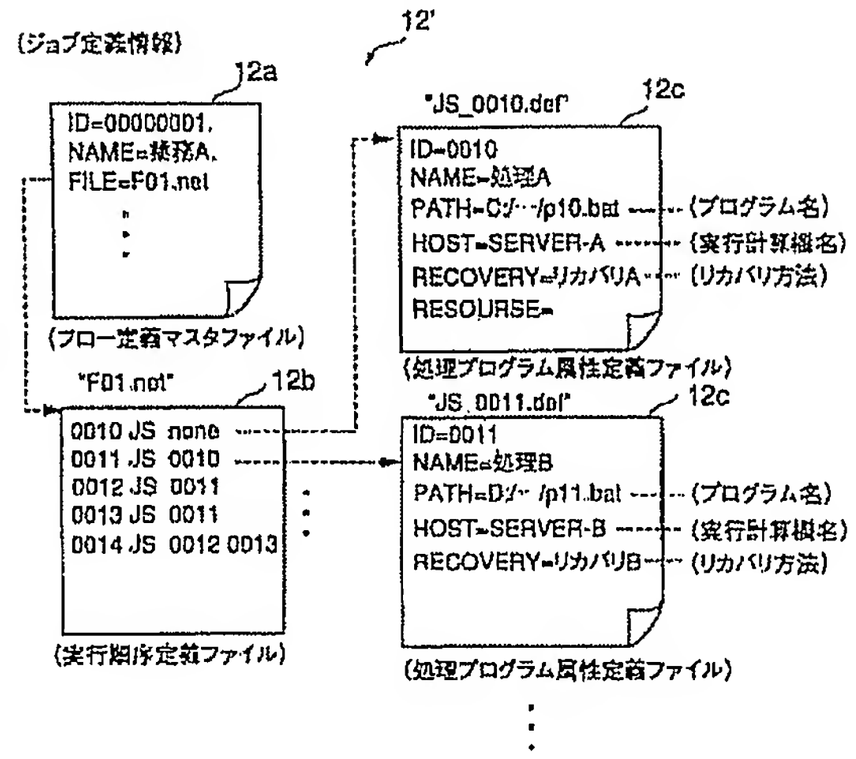
２０　ネットワーク

２２　ジョブ実行要求受付部

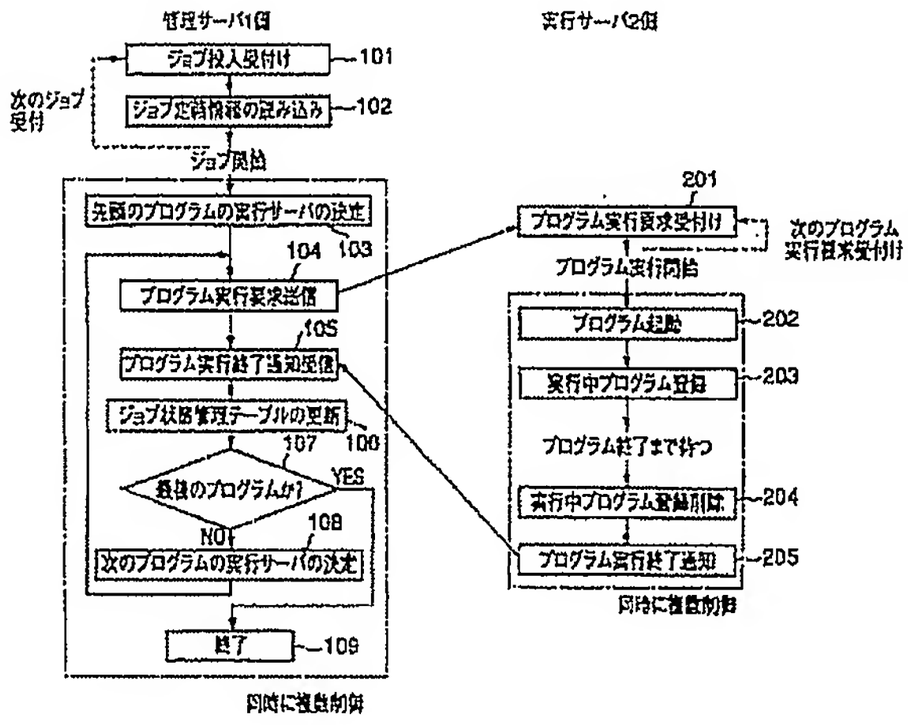
【図1】



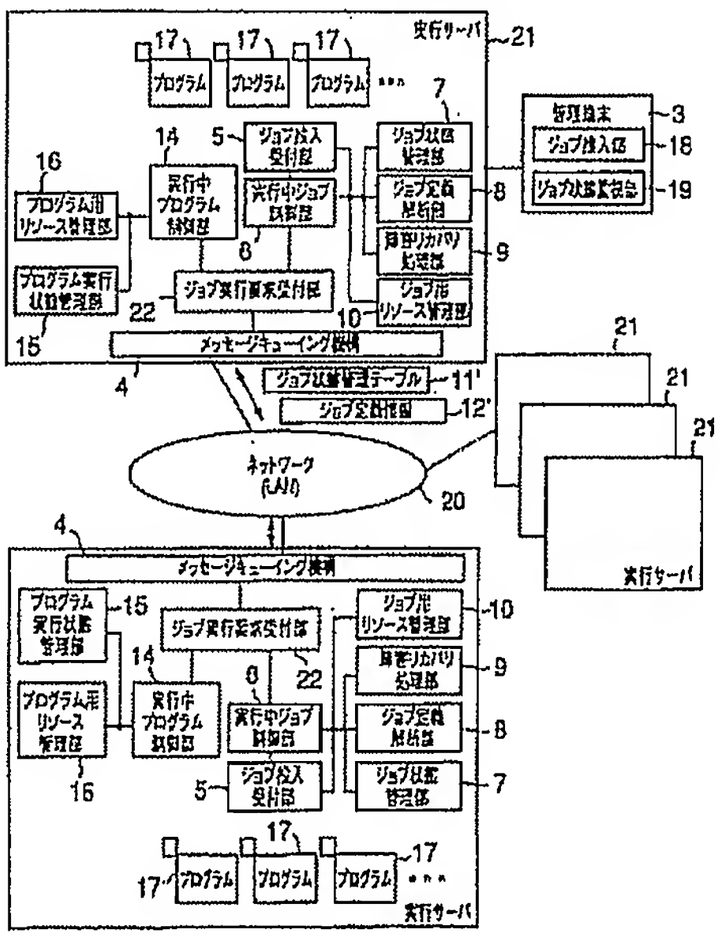
【図3】



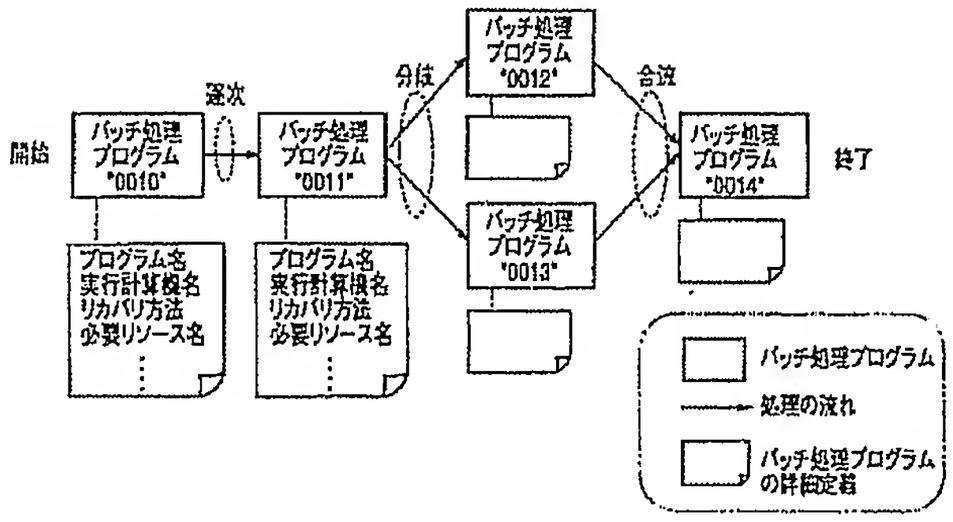
【図2】



【図6】



【図4】



【図5】

(ジョブ状態管理テーブル)　　11'

ジョブID	業務名	投入日時	投入ユーザ	開始日時	終了日時	現在のステータス	実行中ジョブステップ
00000001	業務A	1999/01/01 22:00:00	admin	1999/01/01 22:10:00	1999/01/01 22:30:00	受付中 JS実行待ち JS実行中 終了(正常) 終了(エラー) タイムアウト発生 強制終了 停止中 リカバリ処理中	0010 0011

プログラムID	処理名	投入日時	実行開始日時	実行終了日時	実行計算機	終了のステータス
0010	処理A	1999/01/01 22:10:00	1999/01/01 22:12:00	1999/01/01 22:30:00	SERVER-A	SS00(正常終了) SS01(エラー) SS02(タイムアウト) nnnnアプリ終了コード

フロントページの続き

(51)Int.Cl.[7]	識別記号	FI		テーマコード(参考)
G06F 15/177	678	G06F 15/177	678C	